221-TP-005-001

Treatment of Metadata within the EOSDIS Core System Architecture

Technical Paper

April 1996

Technical Paper--Not intended for formal review or government approval.

Prepared Under Contract NAS5-60000

RESPONSIBLE ENGINEER

Ron Williamson /s/	4/24/96
Ron Williamson, Research Department Manager EOSDIS Core System Project	Date

SUBMITTED BY

Stephen Fox /s/	4/24/96
Steve Fox, TSO Manager EOSDIS Core System Project	Date

Hughes Information Technology Systems
Upper Marlboro, Maryland

This page intentionally left blank.

ECS Document Ref: 221-TP-005-001

Treatment of Metadata within the EOSDIS Core System Architecture

Ron Williamson, Hughes Research Laboratory
301-925-0340, william@eos.hitc.com

Graham Bland, Earth Observation Science Ltd.

ABSTRACT

The EOSDIS Core System (ECS) architecture attempts to remove the arbitrary distinction present in many systems between different levels of metadata, and between metadata and data. Historically, the distinction between data and metadata has been determined by a hard-and-fast rule: metadata is small, and searchable, hence stored in a relational database; data is large, and generally treated as a single object, not necessarily further decomposed in a search. However, this can lead to significant problems in a system as large and long-lived as ECS. By treating metadata and data as logically equivalent entities, we simplify the data access model (there’s a consistent way to get to all data), and provide implementation-transparent data access to users. This frees us up to represent data as needed to support anticipated operations, without making that representation apparent to the user. This approach eliminates the impact in traditional systems of data “moving across the boundary”. Such a reorganization can often have far reaching effects, in the reformatting of databases, and the modification of tools that provide access to the data.

1.0 Introduction

NASA's Earth Observing System (EOS) Core System will be the cornerstone of an new era of multi-disciplinary research to study the processes leading to global climate change. The EOS series of remote-sensing satellites will scan the Earth for several years transmitting hundreds of gigabytes of valuable data every day by the middle of 1999. The giga-flop level of processing and data product generation will require archives to store terabytes of data daily. Over the lifetime of the EOS, the archives will store petabytes, 10^{15} bytes, of data.

The EOS Data Information System (EOSDIS) will provide the computing and network facilities to support the EOS community's research activities. EOSDIS will be an integrated system that supports multiple satellites and instruments, including instruments to be launched by NASA, the European Space Agency (ESA), and the Japanese National Space Agency (NASDA). The system will provide end-to-end services from command and control of spacecraft instrumentation, to data collection and processing, to full access of EOS and other data holdings.

The EOSDIS Core System (ECS), the EOSDIS' infrastructure, will provide scientists a broad range of desk top services for data search and access. According to current plans, the data from each EOS instrument will be sent to one of several designated data centers, known as Distributed Active Archive Centers (DAAC), responsible for processing, archiving, and distributing EOS and related data. NASA selects DAACs based on their expertise in specific science disciplines and demonstrated long-term commitments to the corresponding user communities. These data centers will house the ECS computing facilities and operational staff needed to produce EOS Standard Products and to manage, store, and distribute EOSDIS data and associated metadata. The DAACs will exchange data via dedicated EOSDIS networks to support processing at one DAAC or Science Computing Facility (SCF) requiring data from another DAAC or SCF. With an evolving system, associated DAACs will be added to provide discipline specific key functionality.

2.0 ECS Architecture

ECS provides core reusable functionality to the overall EOSDIS system. As shown in Figure 2-1, the ECS provides Flight Operations support, Science Data Processing, and Communications and Management Services. These are the key segments of the distributed user system. The DAACs receive the EOS level 0 data directly from the EOS Data Operations System (EDOS) and produce the higher level products used by the ECS client-server system. Telemetry and instrument commands link to the Flight Operations segment from the EDOS, while additional data products and algorithms flow into the Science Data Processing segment at each DAAC from Science Computing facilities and Affiliated Data Centers. The ECS Communications and Management Services help coordinate these processes. Together, these three segments work together to insure that users are receiving complete, up-to-date access of all available EOS data products.

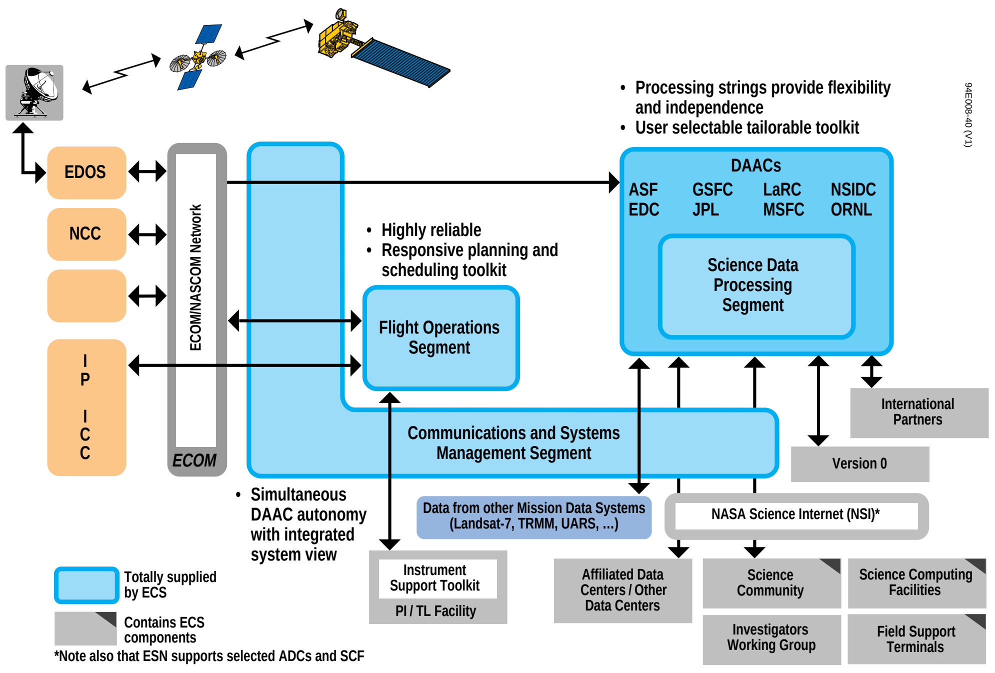
• Processing strings provide flexibility and independence
• User selectable tailorable toolkit
94E008-40 (V1)
EDOS
NCC
IP ICC
ECOM/NASCOM Network
ECOM
DAACs
ASF EDC
GSFC JPL
LaRC MSFC
NSIDC ORNL
Science Data Processing Segment
• Highly reliable
• Responsive planning and scheduling toolkit
Flight Operations Segment
Communications and Systems Management Segment
International Partners
Version 0
• Simultaneous DAAC autonomy with integrated system view
Data from other Mission Data Systems (Landsat-7, TRMM, UARS, ...)
NASA Science Internet (NSI)*
Instrument Support Toolkit
PI / TL Facility
Affiliated Data Centers / Other Data Centers
Science Community
Science Computing Facilities
Investigators Working Group
Field Support Terminals
Totally supplied by ECS
Contains ECS components
*Note also that ESN supports selected ADCs and SCF

Figure 2-1: EOSDIS Core System Context

2.1 Conceptual Architecture

ECS uses an architecture that provides users direct access to data and information as well as interfaces to other services. This arrangement is similar to that of the World Wide Web (WWW) where users access web sites (servers) by way of browsers (clients), such as Netscape or Mosaic, on their computers. Figure 2-2 shows the ECS conceptual architecture.

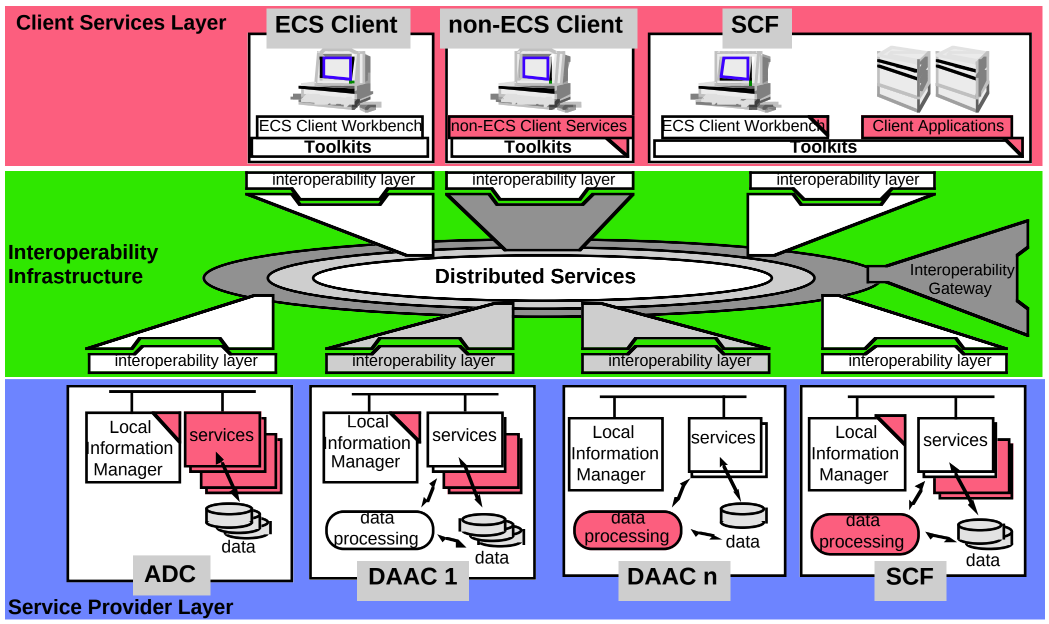

Figure 2-2: ECS Conceptual Architecture

The Client Services layer supports the diverse user community which desires access to EOS data. Users are expected to come from a very large educational institution base, government facilities and even commercial institutions expressing a need to obtain and process EOS data for scientific, commercial and policy making reasons.

The extensible client environment, which is the user interface residing on the user's computer, includes all the tools for data search and retrieval. For example, users can locate and invoke an EOSDIS-wide service, such as cross-DAAC searching, or can choose local DAAC services, such as search of local data connections and DAAC-unique subsetting. A user can also query specific provider services directly, using science-oriented forms or free-text.

The extensible provider network, the interoperability layer, handles the interface between ECS client tools, user driven tools through Application Programming Interfaces (API), and any of the lower level, distributed ECS services. The interoperability layer acts as the "middleware" allowing the client layer to communicate with any or all of the service providers in a seamless and transparent manner.

The evolvable data management and autonomous provider sites compose the service provider layer in this architecture layout. This includes the DAACs, the SCFs, and any other external data provider, from individual scientists to whole data centers. It is at the these lower level layers in the ECS architecture where the active data providers supply their own special data products and services for distributed access. ECS is being designed to allow for a very open ended service provider layer, which is expected to grow over time as the complexity and needs of the program increase.

2.2 Functional Architecture

The functional architecture grouped by major sub-system is illustrated in Figure 2-3. From a Data Center system context view, the various sub-systems comprise the total client/server system. From a user's perspective, the sub-system functionality can be via a set of scenarios showing how data can be 'pushed' by algorithms or external interfaces or 'pulled' by the query or user requests. This approach allows for a better understanding of how end user scenarios work within the system. For instance, the Client sub-system is on the pull side of the sub-system architecture. It allows users to generate queries through the Data Management, Management and Interoperability sub-systems. On the push side, the Planning, Data Processing and Ingest sub-systems deal with the deluge of data and metadata being entered into the system. The Data Server sub-system can be viewed as the central engine to the sub-system architecture. It has links to all of the other sub≠ systems and is chiefly responsible for the handling of all data and services requests.

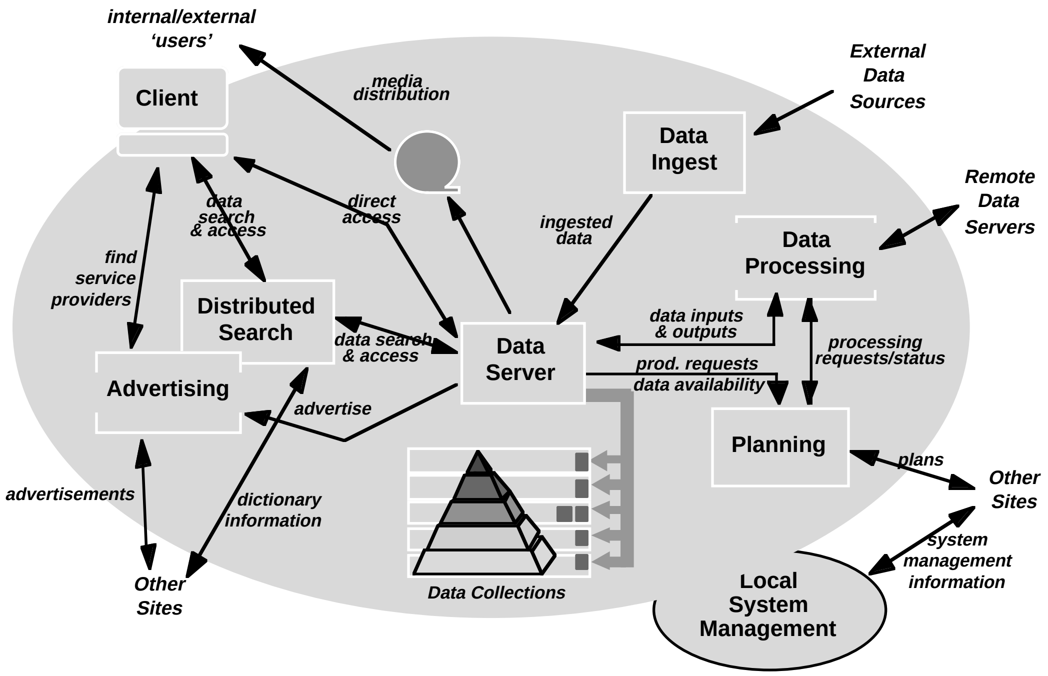

Figure 2-3: Data Center System Context

2.3 Data Architecture

Another critical requirement helping to drive the ECS system architecture is the EOSDIS Data Model. The data model can be viewed as a pyramid, with the lowest level data (data products) at the bottom, and the highest level information, metadata (information about data), at the top. The pyramid model is meant primarily as a conceptual user view of the information and data in ECS and facilitates communication of the domain specific aspects of the data. A major design driver for the data architecture is to blur the distinction between the data and the associated metadata. In some domains the research scientist may wish to treat the product data, such as Digital Elevation Models, as metadata to be used to process other related data products. In other cases the metadata itself is important as a data product in and of itself. The ECS design provides a uniform view of the each of the pyramid layers, providing equivalent search, access, transfromation, and distribution services for metadata and data alike.

Starting at the bottom of the pyramid, EOSDIS data products begin with the EOS level 0 data. From here, higher products are built driving the need for levels 1 through 4. Moving up the pyramid, the levels of data go from less structured to highly structured, from single products to multiple products, from raw units to scientifically generated units and parameters.

The top layers of the pyramid are the higher level services that summarize and supplement the data products. These include information regarding the data production histories, summary statistics, and full-level metadata summaries. Within a data collection can be one or more data granules. A granule is defined as the lowest level data item produced by the data provider. ECS data collections will be driven by the EOSDIS Science Data Plan, which will involve multiple platforms and instruments, and produce multiple types of data: swaths, images, time series data, gridded data, and point data.

ECS must provide access to all levels of the pyramid. This is why the data model, in itself, is one of the biggest design drivers. The ECS software and hardware must be built for the sole purpose of allowing users access to all of the pyramid levels of data and information. In order to better represent this, ECS has broken down its top level system design into sub-systems. Each sub≠ system can be thought of as representing a portion of the functionality needed to get access to the pyramid information. Each sub-system carries with it the necessary structure to carry this information across the communications infrastructure.

The Data Management subsystem provides distributed search and access services with a science≠ oriented view of data collections. There are three levels at which data access requests can be processed: at the intersite level by the Distributed Information Manager (DIM); at the site level by the Local Information Manager (LIM); and at the data set or data-type level by a data server.

The DIM is an example of a service that is capable of executing requests requiring access to multiple sites. The DIM acts as an agent for the client: after the DIM accepts a distributed query or access request, it assumes responsibilities for its execution and for compilation of the results; the client can disconnect from the DIM, reconnect at a later point to determine the status of a query, obtain partial results, or cancel the query.

The LIM is an instance of a service that is capable of executing requests which require access to multiple data servers at a single site. LIM services can be requested by a DIM or directly by a client. In either case, the LIM will act as an agent of its client, just as the DIM. Note that a site may choose to provide several LIMs, perhaps supporting different data access and query languages.

A key element of the Data Management subsystem is the use of an Earth science query language. Initially the language will be based on a set of existing standard query parameters supporting access to a complete collection of Earth science data. This will give the users an Earth science≠ based view of the data instead of a computer science-based view of the data, with which they may be unfamiliar. As the capabilities of the data servers and the LIMs improve, the language will be able to support more complex forms of data searching including coincident search and content≠ based search.

2.4 Integrated Object Architecture

The ECS services-based architecture serves a wide range of user needs and allows users to focus on global change research rather than computer science details. The Client services allow users to easily explore and locate provider services that are advertised through the Interoperability layer. Advertisements provide complementary, coupled views of services, data sets, and providers. After identifying a service of interest, users can immediately invoke the provider service, or the reference to the provider service can be saved on the desktop for later use. For example, users can locate and invoke an EOSDIS-wide service, such as cross-DAAC searching or choose local DAAC services, such as the search of local data collections and DAAC-unique subsetting. A user can also query directly, using science oriented forms or free text, for specific provider services.

Using an object-oriented design methodology, the ECS design is iteratively refined with progressively more definition of sub-systems, lower level objects, and associated interfaces. Evolutionary upgrades will be implemented depending on technology and cost considerations. For example, content-based searching of EOS products is not practical with current technology and is a candidate for future evolutionary upgrade. Other candidate future technologies for insertion will include: RISC processors, parallelization software, optical tape, parallel databases, OODBMS, distributed computing, direct broadcast, Java, and smart agents; some of which are already being prototyped.

3.0 ECS Design Drivers

One of the chief challenges facing ECS is the information system paradigm shift. Traditional distributed information systems are giving way to larger Federated information systems. This means there will be a move from locally distributed systems to globally distributed, homogeneous to heterogeneous system architectures, centrally managed to autonomous, relatively static to dynamic, data searching to information discovery, and standard query interfaces being replaced with hypermedia browse paradigms, automatic subscriptions and intelligent agents. Given this evolutionary constraint, ECS must design with change in mind.

Over the twenty year life-span of EOSDIS, the dual system drivers of cost reduction pressures and evolution will come from at least three distinct sources: 1) Scientific needs will change as Earth science matures with new data products and new applications mature; 2) Information system technologies must be refreshed as maintaining older technologies becomes more costly and newer technologies displace them; and 3) Changes in information infrastructure (i.e., high bandwidth networking) will lead to migration of, extension to, and the addition to the core functions of EOSDIS. While predicting two decades into the future is an almost impossible task to perform accurately, NASA and the Hughes Team have taken aggressive steps to reduce overall life-cycle costs and preserve a framework for evolutionary enhancement of the system.

The science drivers are dictated largely by the research scientists' needs in supporting long term Global Change research. They are centered around the needs of research teams to effectively and efficiently find, access, use, and share the results of their research, which include data, modeling, and observations and analyses. The science drivers are derived from various inputs from the scientific community, including feedback from design reviews and visits with research teams associated with Science Computing Facilities (SCFs). They include:

1. Facilitate an efficient data search and "access" paradigm
2. Support a dynamic product lifecycle and easily extensible product set
3. Support an interactive investigation capability
4. Support an information-rich data pyramid
5. Support the integration of independent investigator tools
6. Support user-to-user collaboration
7. Provide distributed administration and control

Each of these science drivers were analyzed and mapped into a set of key design drivers for the ECS Architecture. Three of these design drivers, critical to the design of ECS metadata and data architectures, are described next.

3.1 Data Equals Metadata

This concept attempts, among other things, to remove the arbitrary distinction present in many systems between data and metadata. Historically, the distinction between data and metadata has been determined by a hard-and-fast rule: metadata is small, and searchable, hence stored in a relational DB; data is large, and generally treated as a single object, not necessarily further decomposed in a search. However, this can lead to significant problems in a system as large and long-lived as ECS, e.g.:

- one person’s data is another person’s metadata

  For example, masks can be viewed simply as image overlays, used in compositing for display purposes. Alternatively, they might be used in spatial relation operators (i.e. "where") to subselect data based on element-wise mask values. In this latter case, the data values within the mask "image" are actually used in element-wise relational operations to subset another image.

- metadata changes over time

  Over time, "data" may be re-characterized in a number of different ways (i.e. the metadata descriptions may change), for example as more is learned about a particular sensor's data. (re-characterize data, etc.). The ability to redefine what conceptually amounts to metadata over time is important for long term viability of datasets.

By treating metadata and data as logically equivalent entities, we simplify the data “access” model (there’s a consistent way to get to all data), and provide implementation-transparent data access to users. This eliminates the impact in traditional systems of data "moving across the boundary". Such a reorganization can often have far reaching effects, in the reformatting of databases, and the modification of tools that provide access to the data.

3.2 Data Equals Services

A similar concept is applied to treat data and services identically. This is important because the optimal manner for providing data to users may be dataset and product level specific, and may change over time. For example, it may be "cheaper" to create a level 3 data product directly from its base level 1 product each time it is requested, rather than pre-computing and storing level 2 and 3 products. This would be true if, for example the algorithm for producing the level 3 product were changing frequently and the requests for that product were infrequent.

By treating data and services as logically equivalent entities, we (again) simplify the data “access” model, because there’s a consistent way to get to all data, whether it’s stored or computed. And, importantly, we support the development of methods which can be used in a common manner to locate both data and system services -- so search methods can be applied uniformly across data and services.

This concept is important in preserving transparency regardless of whether data is retrieved from archive, retrieved from a traditional database, or computed “on-the-fly”. It allows process vs. store trades to be made independently across datasets (or portions of datasets), and over time. The trade can then respond directly to economic and anticipated use constraints, and not be burdened with complex architectural implications.

3.3 Universal Reference

A *Universal Reference* provides a universal way of identifying objects in the system. The syntax and meaning of a universal reference follow standard conventions which every component in the network recognizes and supports. It encapsulates the identifiers which are used internally at a site or by a specific service, providing a system-wide token that can be successfully exchanged for the referenced data from any location within the system.

Some key features of the universal reference concept that make it suitable for the needs of ECS include:

- the need to defer retrieval of information until it is used

  By providing a "handle" to data objects, a universal reference allows search and analysis tools to provide users with summary information ("metadata") about data objects, without actually sending potentially large data objects themselves. The user can examine the summary information, perhaps browsing through reduced resolution images of interest, before selecting the data to be obtained for further analysis or delivery. When the user wants to actually retrieve the full resolution imagery, he can do so through a "de referencing" process, in which the data is instantiated and delivered.

- the ability to collect and use "like" items from multiple heterogeneous providers

  The use of universal references also allows collections of "like" objects to be developed and maintained irrespective of source representation of the data. Hence data "objects" that reside in the archive and data "objects" that are computed on demand can be treated equally through universal reference handles. The act of accessing the data then appears identical to the user, regardless of the underlying provider's implementation.

- the ability to easily exchange information with colleagues

  Finally, the universal reference provides a means for researchers to easily exchange information with colleagues. Rather than having to email large satellite images, for example, collaborating researchers can pass references to the data objects. The universal nature of these references guarantees the ability to instantiate or de reference the object from anywhere within the ECS system (and even beyond, assuming "gateway" support to convert the reference into a suitable data stream).

4.0 ECS Object Model

The ECS object model is defined at three levels of abstraction: conceptual, logical, and physical (Data Formats). The data pyramid , Figure 4-1, is the starting point for the conceptual modeling and analysis of ECS data. The layers in the data pyramid are described as a user oriented conceptual model and relate to widely understood groupings of metadata attributes and services as well as various types of products and other system inputs and output. The pyramid layers form a subset of the conceptual clustering of the data into data types, also known as Earth Science Data Types (ESDTs). These layers provide a user view of the data and contain attributes, services, and other descriptive information. The more traditional approach to these data is to separate data (products) from metadata which provide access keys. However, the process of analyzing the pyramid as well as experience, shows that implementing a system which acknowledges interplay between layers is critical to providing a comprehensive data information system.

The process of collecting and analyzing requirements lead to the generation of a data model. This model consists of attributes grouped into a number of objects in seven modules. The model supports the access functionality between pyramid layers by describing at a conceptual level the relationships between data objects. Each of the conceptual model elements, ESDTs, are mapped onto the logical representation defined by a set of computer science data types (CSDTs) as listed in Figure 4-2. The logical level of abstraction, captured by the CSDT's, allows the mapping of the ESDT's onto a structure that can then be implemented by COTS data management and access systems. For example, the Directory and Inventory are completely characterized both structurally and functionally by an "access table" data type that is mapped onto two different database management systems, Sybase and Illustra.

The physical level of abstraction captures the mapping of the data onto physical media through the definition of a set of standard data formats. There have been several attempts to develop standard data formats for earth science data (e.g. SFDU, netCDF, FITS, BUFR, HDF, etc.). For EOSDIS the HDF standard has been chosen. The pragmatic earth observation model of providing data in the form of 'products' with a single defined format, however, is not ideal for many research uses and can be difficult to apply to many data sets. Inevitably, such products are a compromise between the various applications which may utilize the data in such characteristics as granularity, gridding and even processing algorithms. For example, a provider might wish to store an altimeter data set in a time ordered fashion with each granule related to an orbit revolution for simple archiving from a continuous processing scheme. This data structure is clearly not efficient for studies of local geographic areas such as areas of extreme wind events, and ice shelf elevation, and the provider may therefore choose to offer a service which permits several views of the data, and even allows the dynamic reformatting of data products to match the different views presented. Within the ECS context, the provider (DAAC, SCF, or other value added provider) may declare different views and formats for a single dataset and the user will be able through the request process to select the view and data format required.

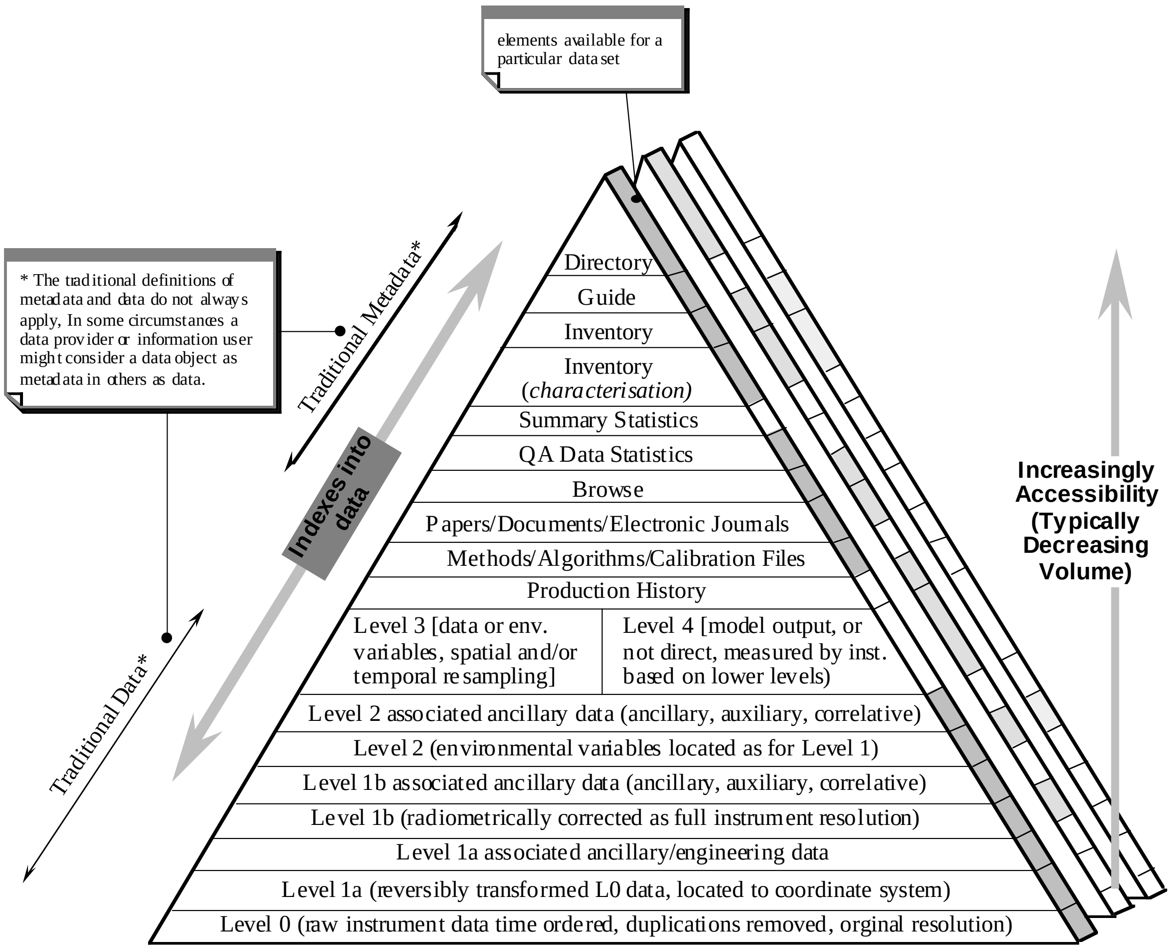
elements available for a particular data set
* The traditional definitions of metadata and data do not always apply, In some circumstances a data provider or information user might consider a data object as metadata in others as data.
Traditional Metadata*
Traditional Data*
Indexes into data
Directory
Guide
Inventory
Inventory (*characterisation)*
Summary Statistics
QA Data Statistics
Browse
Papers/Documents/Electronic Journals
Methods/Algorithms/Calibration Files
Production History
Level 3 [data or env. variables, spatial and/or temporal resampling]
Level 4 [model output, or not direct, measured by inst. based on lower levels)
Level 2 associated ancillary data (ancillary, auxiliary, correlative)
Level 2 (environmental variables located as for Level 1)
Level 1b associated ancillary data (ancillary, auxiliary, correlative)
Level 1b (radiometrically corrected as full instrument resolution)
Level 1a associated ancillary/engineering data
Level 1a (reversibly transformed L0 data, located to coordinate system)
Level 0 (raw instrument data time ordered, duplications removed, orginal resolution)
Increasingly Accessibility (Typically Decreasing Volume)

Figure 4-1 The Data Pyramid

Pyramid layers	Layers/ESDTs	CSDT
Metadata	Directory	
	Inventory	Access table.
	Inventory characterization	
Auxiliary	Guide	
Metadata	Summary statistics	Documents
	QA data statistics	Code
	Reference papers	N-Dimensional Array
	Delivered algorithm package	Science Data Table
	Production history	Image.
	Browse products	
	Non-geolocated ancillary data	
Products		Grid, Swath, Point
	Level 1,2,3,4 data	N-Dimensional Array
	Level 0 data	Science Data Table
	Engineering data	P=V Metadata
	Geolocated ancillary data	ASCII Text.

Figure 4-2 The ECS Data Architecture: ESDT's and CSDT's

5.0 References

Bland, Graham, Preliminary User View of Release A Data, Document # 420-TP-005-002, August, 1995.

Dopplick, Tom, A Science User's Guide to the EOSDIS Core System (ECS) Development Process, ECS Document # 160-TP-003-001, February, 1995.

ECS, Summary of the ECS System Design Specification. http://edhs1.gsfc.nasa.gov, 1994

ECS, "Science-based System Architecture Drivers for the ECS Project" whitepaper, ECS Document # 193-00611, December, 1993.

ECS, "ECS Evolutionary Development White Paper", ECS Document # 193-00623, December, 1993.

Elkington, M., Meyer, R. and McConaughy, G., Defining the Architecture of EOSDIS to Facilitate Extension to a Wider Data Information System. ISPRS'94, Ottawa, 1994.

Elkington, M, Meyer, R., "GCDIS / UserDIS Study" whitepaper, ECS Document # 193-000626, January, 1994

Fox, Steve, EOSDIS Core System Science Information Architecture, ECS Document # FB9401V2, March, 1994.

Heller, Denise, ECS Technical Notes, Query Classes, March, 1994.

Heller, Denise, Proposed ECS Core Metadata Standard - Release 2.0 , ECS Document # 420-TP-001-005, December 1994.

Hylton, Janet, SDPS Database Design and Database Schema Specifications, ECS Document # 311-CD-002-004, December, 1995.

Moxon, B. 1993: 19300611 Science-based System Architecture Drivers for the ECS Project. URL: http://edhs1.gsfc.nasa.gov.